Syddansk Universitet

Parametric model reduction via interpolating orthonormal bases

Zimmermann, Ralf; Debrabant, Kristian

Published in:
Numerical Mathematics and Advanced Applications

Publication date:
2018

Document version
Peer reviewed version

Citation for pulished version (APA):
Zimmermann, R., & Debrabant, K. (2018). Parametric model reduction via interpolating orthonormal bases. In Numerical Mathematics and Advanced Applications: ENUMATH 2017 Springer. Lecture Notes in Computational Science and Engineering

Parametric model reduction via interpolating orthonormal bases

Ralf Zimmermann[1] and Kristian Debrabant[1]

University of Southern Denmark, Campusvej 55, 5230 Odense M, DK,
zimmermann@imada.sdu.dk, debrabant@imada.sdu.dk

Abstract. In projection-based model reduction (MOR), orthogonal coordinate systems of comparably low dimension are used to produce ansatz subspaces for the efficient emulation of large-scale numerical simulation models. Constructing such coordinate systems is costly as it requires sample solutions at specific operating conditions of the full system that is to be emulated. Moreover, when the operating conditions change, the subspace construction has to be redone from scratch.

Parametric model reduction (pMOR) is concerned with developing methods that allow for parametric adaptations without additional full system evaluations. In this work, we approach the pMOR problem via the quasi-linear interpolation of orthogonal coordinate systems. This corresponds to the geodesic interpolation of data on the Stiefel manifold. As an extension, it enables to interpolate the matrix factors of the (possibly truncated) singular value decomposition. Sample applications to a problem in mathematical finance are presented.

1 Introduction

Model reduction (MOR) is a branch of applied mathematics that is concerned with the emulation of large-scale dynamical systems via a highly reduced number of degrees of freedom. The resulting reduced model (ROM) is expected to be much faster to evaluate, but less accurate than the original model. Ideally, it comes with inherent error indicators/estimators that allow the user to control the trade-off between the numerical efficiency and the numerical accuracy.

Subspace-based model reduction. Among the most prominent model reduction techniques are projection-based methods. Here, the key idea is to construct a low-dimensional subspace of *solution candidates* and to restrict all successive computations to this subspace, e. g. via a projection.

We explain this procedure on an example. A full survey of methods can be found in [2]. Consider a spatio-temporal dynamical partial differential equation (PDE) system in semi-discrete form

$$\frac{\partial}{\partial t} y_q(t) = f(t, y_q(t)), \quad y_q(t) \in \mathbb{R}^n, \quad f : [t_0, t_e] \times \mathbb{R}^n \to \mathbb{R}^n.$$

Here, we assume that the spatial dimensions are resolved in the discretization with a grid of $n \in \mathbb{N}$ points. The index q indicates additional system parameters $q = (q_1, \dots, q_d)^T \in \mathbb{R}^d$ on which the system may depend.

Subspace construction is based on *sampled snapshot solutions* at m selected sample points: $y^1 = y_q(t_1), \dots, y^m = y_q(t_m)$. It is assumed that the system dimension n exceeds the number of sampled snapshots m by several orders of magnitude, $n \gg m$. The subspace of solution candidates is to represent the span of the sampled snapshots but truncated to the essential irredundant information. This is achieved via a *compact singular value decomposition (SVD)*, which corresponds to the most basic form of a proper orthogonal decomposition [9] of the input data,

$$U\Sigma V^T \overset{SVD}{=} Y := (y^1, \dots, y^m), \tag{1}$$

with $U \in \mathbb{R}^{n\times m}, V \in \mathbb{R}^{m\times m}, \Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_m)$, followed by a truncation of U, V to the first $p \leq m$ columns that are associated with the most significant non-zeros singular values. A popular measure for this notion of significance is the *relative information content*, $RIC(p) = \frac{\sum_{j=1}^{p} \sigma_j}{\sum_{k=1}^{m} \sigma_k}$. In practice, p is chosen according to a user-defined threshold ε such that $RIC(p) \geq 1-\varepsilon$. Perfect recovery of the snapshot data (up to numerical errors) corresponds to $\varepsilon = 0$. The process of snapshot sampling and basis construction is referred to as the *off-line stage* of the ROM and is assumed to be computationally intense, since solutions to the original system are required.

The truncated $U \in \mathbb{R}^{n\times p}$ can be interpreted as a *coordinate system* for solution candidates for the ROM: $\tilde{y}_q(t) = U\hat{y}_q(t)$, where $\hat{y}_q \in \mathbb{R}^p$ is the vector of basis coefficients with respect to the coordinate system induced by U and $\tilde{y}_q$ is the ROM approximation of the exact solution y_q. One possible way to obtain the reduced coefficient vector $\hat{y}_q$ is via Galerkin projection of the original system onto the reduced coordinates:

$$\frac{d}{dt}\hat{y}_q(t) = U^T f(t, U\hat{y}_q(t)). \tag{2}$$

Yet, several other approaches to determine the coefficient vector $\hat{y}_q(t)$ exist, including Petrov-Galerkin projection, the discrete empirical interpolation method (DEIM) [6], interpolation [4,11] and residual optimization [8,5,14]. The process of determining $\hat{y}_q(t)$ is referred to as the *on-line stage* of the ROM and is designed such that it is independent of the original system dimension n or scales at most linear in n.

Problem statement. The main focus of this work is not on approaches to determine the reduced state $\hat{y}_q(t)$ but on the variation of the ROM itself under changes of a scalar system parameter $q \in \mathbb{R}$. We say that q specifies the *operating condition*. Our objective is to move from q to $\tilde{q}$ without repeating the off-line process of snapshot sampling and coordinate system construction. To this end, we parameterize this process via interpolation, but on the level of the orthogonal bases that define the ROM candidate solution subspaces, where we assume that the snapshot matrix $q \mapsto Y = Y(q)$ depends differentiably on the operating condition q. Consider two snapshot matrices $Y(q_0)$

and $Y(q_1)$ with possibly truncated SVDs

$$U(q_0)\Sigma(q_0)V^T(q_0) \stackrel{SVD}{\approx} Y(q_0), \quad U(q_1)\Sigma(q_1)V^T(q_1) \stackrel{SVD}{\approx} Y(q_1), \qquad (3)$$

where the approximation holds up to a specified relative information content. More precisely, we assume that the reduced subspace dimension p is chosen such that $RIC(p) \geq 1 - \varepsilon$ *at both operating conditions* q_0, q_1. In particular, we require that $U(q_0)$ and $U(q_1)$ share the same dimensions.

From this perspective, the task to construct a *parametric ROM* essentially reduces to the task of computing a trajectory $q \mapsto U(q)$ that starts in $U(q_0)$ and ends in $U(q_1)$ *without* having to enter a new off-line stage for every q.

The main obstacle is that this trajectory is a curve in the set of orthogonal coordinate systems. This set forms a curved matrix manifold, the so-called *Stiefel manifold* [1,7],

$$St(n,p) = \{U \in \mathbb{R}^{n\times p} | \quad U^T U = I_p\}.$$

Our original contribution is a method for quasi-linear geodesic interpolation on the Stiefel manifold.

Interpolation procedures on matrix manifolds have been considered for parametric model reduction before, see [2, §4] and references therein. The standard technique is to (1) first map the matrix data onto the flat tangent space of the manifold, (2) perform the interpolation in the tangent space, (3) map back the result to the matrix manifold. This procedure is illustrated in Fig. 1. However, to the best of our knowledge, interpolation of data on

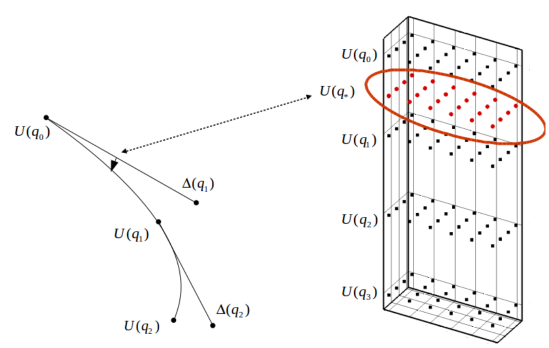

Fig. 1. Interpolation of orthogonal bases. The curved line represents the Stiefel manifold; the straight lines represent the tangent vectors in the tangent space at $U(q_0)$ and $U(q_1)$, respectively.

the Stiefel manifold has not yet been treated in the literature. A partial explanation is that conducting the back-and-forth mapping between manifold data and tangent vectors requires a practical method for computing both the

Riemannian exponential [1, §5.4] on the Stiefel manifold and its inverse, the Riemannian logarithm. While an explicit formula for computing the exponential on the Stiefel manifold exists for almost two decades [7], an efficient algorithm for computing the Riemannian logarithm has only recently been developed [10,13].

2 The Stiefel manifold in numerical schemes

In this section, we recap the essentials of working with data on the Stiefel manifold in numerical procedures. For more details, the reader is referred to [1,7]. The *tangent space* $T_U St(n,p)$ at a point $U \in St(n,p)$ is represented by

$$\begin{aligned} T_U St(n,p) &= \left\{ \Delta \in \mathbb{R}^{n\times p} | \quad U^T \Delta = -\Delta^T U \right\} \\ &= \left\{ UA + (I - UU^T)T | \quad A \in \mathbb{R}^{p\times p} \text{ skew}, T \in \mathbb{R}^{n\times p} \right\} \subset \mathbb{R}^{n\times p}. \end{aligned}$$

The Riemannian exponential $t \mapsto Exp^{St}_{U_0}(t\Delta)$, which gives the geodesic curve starting at $t = 0$ in U_0 with velocity Δ can be computed with the standard matrix exponential as a building block: Let $QR = \Delta$ be the (compact) QR-decomposition of the tangent vector, then

$$\tilde{U} = Exp^{St}_{U_0}(\Delta) \;=\; U_0 M + QN \in St(n,p), \text{ where} \tag{4a}$$

$$\begin{pmatrix} M \\ N \end{pmatrix} := \exp_m\left(\begin{pmatrix} A & -R^T \\ R & 0 \end{pmatrix} \right) \begin{pmatrix} I_p \\ 0 \end{pmatrix}, \quad A = U_0^T \Delta, \tag{4b}$$

see [7, §2.4.2]. A matrix-algebraic algorithm for computing Δ, given two points $U, \tilde{U} \in St(n,p)$, was introduced in [13]. This scheme produces a sequence of skew-symmetric matrices $A_k \in \mathbb{R}^{p\times p}$ and matrices $R_k \in \mathbb{R}^{p\times p}$ such that in the limit,

$$\Delta = Log^{St}_U(\tilde{U}) = U_0 A_\infty + QR_\infty, \tag{5}$$

where Q stems from the QR-decomposition of $(I - UU^T)\tilde{U}$. For full details and MATLAB code, see [13].

3 Quasi-linear interpolation of orthogonal bases

Assume that we are given two snapshot ensembles $Y(q_0)$, $Y(q_1)$ with SVDs as in (3). With an efficient algorithm for the Riemannian Exp^{St} and Log^{St}, we may perform quasi-linear interpolation on $St(n,p)$ to obtain a trajectory of orthogonal bases $q \mapsto U(q) \in St(n,p)$. To this end, we use $U(q_0)$ as a base point and map $U(q_1)$ to $T_{U(q_0)}St(n,p)$. In this way, we obtain a velocity vector Δ that corresponds to the geodesic on $St(n,p)$ that starts in $U(q_0)$ and eventually crosses $U(q_1)$. Since geodesics on curved manifolds are the natural generalization of straight lines in Euclidean spaces, we interpret this as *quasi-linear* interpolation. The algorithmic procedure is summarized in Alg. 3.1. It may be extended to interpolating the complete SVD, see Alg. 3.2.

Algorithm 3.1 Geodesic interpolation on $St(n,p)$

Input: $U(q_0), U(q_1) \in St(n,p)$, $s \in [q_0, q_1]$

1: $\Delta = Log^{St}_{U(q_0)}(U(q_1))$ %velocity vector, see (5), [13, Alg. 1]

2: $q(s) = \frac{s-q_0}{q_1-q_0}$

Output: $U(q(s)) := Exp^{St}_{U(q_0)}(q(s)\Delta)$ % see (4a), (4b)

Algorithm 3.2 Geodesic interpolation of SVD data.

Input: $U(q_0), \Sigma(q_0), V(q_0), U(q_1), \Sigma(q_1), V(q_1)$, $s \in [q_0, q_1]$

1: $\Delta_U = Log^{St}_{U(q_0)}(U(q_1))$, $\Delta_V = Log^{St}_{V(q_0)}(V(q_1))$, $\Delta_\Sigma = \Sigma(q_1) - \Sigma(q_0)$

2: $q(s) = \frac{s-q_0}{q_1-q_0}$

Output: $Exp^{St}_{U(q_0)}(q(s)\Delta_U) \cdot (\Sigma(q_0) + q(s)\Delta_\Sigma) \cdot \left(Exp^{St}_{V(q_0)}(q(s)\Delta_V)\right)^T$

4 Numerical experiments

In this section, we present an academic application to computational option pricing. The value function $y(T,S;K,r,\sigma)$ that gives the fair price for a European call option is determined via the *Black-Scholes-equation* [3],

$$y_t(t,S) = \frac{1}{2}\sigma^2 S^2 y_{SS}(t,S) + rSy_S(t,S) - ry(t,S), \quad S \geq 0, \quad 0 < t \leq T,$$
$$y_t(0,S) = \max\{S-K, 0\}, \quad S \geq 0.$$

It is a parabolic PDE that depends on time t, the stock value S, also called the underlying, and a number of additional system parameters, namely the strike price K, the interest rate r, the volatility σ and the exercise time T. In this experiment, we consider a fixed interest rate of $r = 0.01$ and an exercise time of $T = 2$ units. The dependency on the underlying $S \in [50, 150]$ is resolved via a discretization of the interval by equidistant steps of $\Delta S = 0.01$, while the strike price $K \in [30, 170]$ is discretized in steps of $\Delta K = 1$. The volatility σ will act as the operating condition. We are interested in approximating the option price y as well as its derivative $\partial_\sigma y$. This quantity is also called 'vega' and belongs to the set of the 'greeks', i.e., to the partial derivatives of the option value function with respect to the system parameters.

The Black-Scholes equation for a single underlying has a closed-form solution. Yet, here, we will approach it via a numerical scheme in order to mimic the corresponding procedure for real-life problems. Application of a finite volume scheme to the Black-Scholes PDE yields snapshot matrices

$$Y(\sigma) = (Y_{s,k}(\sigma))_{\substack{s=1,\dots,10001\\k=1,\dots,141}}, \qquad \partial_\sigma Y(\sigma)_{\substack{s=1,\dots,10001\\k=1,\dots,141}},$$

for $\sigma \in [0, 0.1, 0.2, \dots, 1.5]$. The off-line calculation time amounts to 19+30 min per snapshot matrix (solutions + derivatives). The POD/SVD of a snapshot matrix $Y(\sigma) \in \mathbb{R}^{n\times m}$, $n = 10001$, $m = 141$ yields a compressed representation $Y(\sigma) = U(\sigma)\Sigma(\sigma)V^T(\sigma)$, with $U(\sigma) \in St(n,p)$, $\Sigma(\sigma) \in diag(p,p)$, $V(\sigma) \in St(m,p)$ and consumes ca. $0.07s$ on a laptop computer.

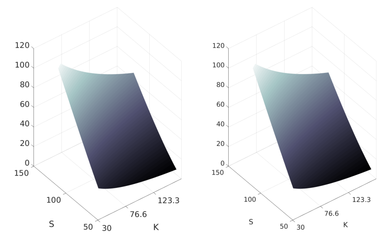

Fig. 2. Left: Batch SVD interpolation of $Y(1.0)$. Right: exact reference.

1^{st} Experiment: We take the snapshot matrices $Y(0.9), Y(1.1) \in \mathbb{R}^{n\times m}$ as sample data. The goal is to compute a low-rank-trajectory $q \mapsto \hat{Y}(q)$ of snapshot matrix approximants. We exemplify this by predicting the full snapshot ensemble $Y(1.0)$ via $\hat{Y}(1.0)$. To this end, we compute the SVDs of $Y(0.9)$, $Y(1.1)$. The original column dimension $m = 141$ is reduced to $p = 5$, which corresponds to a relative information content of $RIC(p) = 1.0 - 10^{-7}$. We interpolate each corresponding SVD matrix factor via Alg. 3.2 in order to obtain interpolants $\hat{U}(1.0)$, $\hat{\Sigma}(1.0)$, $\hat{V}(1.0)$. We assess the accuracy of the approach by the following means:
(1) When we recompose the matrix $\hat{Y}(1.0) = \hat{U}(1.0)\hat{\Sigma}(1.0)\hat{V}(1.0)$, we obtain a relative error of $\|\hat{Y}(1.0) - Y(1.0)\|/\|Y(1.0)\| = 0.00267$ with respect to the reference solution. For comparison purposes, we perform direct, non-compressed linear interpolation of the snapshot matrices, i.e., we compute $Y_{dLI}(1.0) = 0.5(Y(0.9) - Y(1.1))$. The result features a higher relative accuracy of $\|Y_{dLI}(1.0) - Y(1.0)\|/\|Y(1.0)\| = 0.00146$. In fact, we cannot expect that the quasi-linear interpolation of the *compressed data* outperforms the direct linear interpolation of the full data.
(2) When we only consider the interpolated coordinate system $\hat{U} = \hat{U}(1.0)$ and project the full snapshot data $Y(1.0)$ onto the associated subspace, we obtain $\|\hat{U}\hat{U}^T Y(1.0) - Y(1.0)\|/\|Y(1.0)\| = 8.532 \cdot 10^{-9}$. In contrast, when we first compute $Y_{dLI}(1.0)$ and the associated SVD factor U_{dLI} and truncate this to the same dimension as $\hat{U}$, the projection error is $\|U_{dLI}U_{dLI}^T Y(1.0) - Y(1.0)\|/\|Y(1.0)\| = 4.759 \cdot 10^{-8}$, which is roughly 5.6 times higher.

2^{nd} Experiment: Here, we use $\partial_\sigma Y(0.9)$, $\partial_\sigma Y(1.1)$ as sampled input data to predict $\partial_\sigma Y(1.0)$. We repeat the same steps as in the first experiment with $RIC(p) = 1.0 - 10^{-7}$, $p = 7$. For brevity, we state only the results:

$$
\begin{aligned}
(1)\ & \|\partial_\sigma \hat{Y}(1.0) - \partial_\sigma Y(1.0)\|/\|\partial_\sigma Y(1.0)\| && = 0.00380, \\
& \|\partial_\sigma Y_{dLI}(1.0) - \partial_\sigma Y(1.0)\|/\|\partial_\sigma Y(1.0)\| && = 0.00396, \\
(2)\ & \|\hat{U}\hat{U}^T \partial_\sigma Y(1.0) - \partial_\sigma Y(1.0)\|/\|\partial_\sigma Y(1.0)\| && = 1.454 \cdot 10^{-8}, \\
& \|U_{dLI}U_{dLI}^T \partial_\sigma Y(1.0) - \partial_\sigma Y(1.0)\|/\|\partial_\sigma Y(1.0)\| && = 2.410 \cdot 10^{-8}.
\end{aligned}
$$

The results are displayed in Figs. 2, 3. In both experiments, the quality of

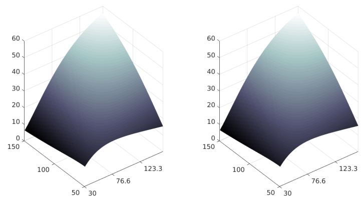

Fig. 3. Left: Batch SVD interpolation of $\partial_\sigma Y(1.0)$. Right: exact reference.

the quasi-linear low-rank SVD interpolation is of the same accuracy order as the direct linear interpolation of the given data matrices. The prediction capabilities of the geodesically interpolated coordinate system are slightly better than for its directly computed counterpart. It is remarkable that even for this rather academic example, performing a single SVD of a snapshot ensemble takes longer than conducting Alg. 3.2 *from scratch*, including the iterative Stiefel log procedure [13, Alg. 1] *and* re-assembling the output matrix ($\sim 0.07s$ vs. $\sim 0.02s$). Note that step 1 of Alg. 3.2 has to be performed only once and that some of the quantities that are required for the Stiefel exponential may also be pre-computed, see [12]. Excluding these operations, the computation time reduces to $\sim 0.007s$, ten times less than for performing the SVD.

5 Summary and conclusion

We propose to use the Riemannian exponential and logarithm mappings to conduct quasi-linear interpolation on the Stiefel manifold. This results in a method for *parametric model reduction* that is completely *data-driven* in the sense that it relies only on output data samples of a given simulation model but does not require any intrinsic system modifications.

The approach can be extended to an interpolation scheme for the singular value decomposition. In this form, it results in a 'batch' method: it directly gives approximations of the trajectory of the full, parameter-dependent snapshot matrices but relies on low-rank quantities exclusively.

Yet, it may also be used as a building block in more sophisticated schemes: We may only interpolate the low-rank coordinate system $U(q)$ and use the corresponding subspace as the space of solution candidates in combination with Galerkin projection, DEIM, residual optimization or other model reduction techniques.

The method leads to a gain in efficiency if the snapshot input data allows for a high level of compression ($p \ll m$). Otherwise, one could directly interpolate the snapshot data matrices and the numerical approximations obtained in this way should be of higher accuracy than when interpolating every matrix factor in a low-rank SVD separately.

References

1. P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, Princeton, New Jersey, 2008.
2. P. Benner, S. Gugercin, and K. Willcox, *A survey of projection-based model reduction methods for parametric dynamical systems*, SIAM Review **57**:4 (2015), 483–531.
3. F. Black and M. Scholes, *The pricing of options and corporate liabilities*, J. Polit. Econ. **81**:3 (1973), 637–654.
4. T. Bui-Thanh, M. Damodaran, and K. Willcox, *Proper orthogonal decomposition extensions for parametric applications in transonic aerodynamics*, AIAA Journal **42**:8 (2004), 1505–1516.
5. K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, *The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows*, Journal of Computational Physics **242**:0 (2013), 623–647.
6. S. Chaturantabut and D. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM Journal on Scientific Computing **32**:5 (2010), 2737–2764.
7. A. Edelman, T. A. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM Journal on Matrix Analysis and Application **20**:2 (1999), 303–353.
8. P. A. LeGresley and J. J. Alonso, *Investigation of non-linear projection for POD based reduced order models for aerodynamics*, Paper 2001-0926, 39th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2001.
9. R. Pinnau, *Model reduction via proper orthogonal decomposition*, Model Order Reduction: Theory, Research Aspects and Applications (W. H. A. Schilders, H. A. Van der Vorst, and J. Rommes, eds.), Springer Series Mathematics in Industry, vol. 13, Springer, 2008, pp. 95–109.
10. Q. Rentmeesters, *Algorithms for data fitting on some common homogeneous spaces*, Ph.D. thesis, Université Catholique de Louvain, Louvain, Belgium, July 2013.
11. R. Zimmermann, *Gradient-enhanced surrogate modeling based on proper orthogonal decomposition*, Journal of Computational and Applied Mathematics **237**:1 (2013), 403 – 418.
12. ———, Local Parametrization of Subspaces on Matrix Manifolds via Derivative Information, 379–387, Springer International Publishing, Cham, 2016, pp. 379–387.
13. ———, *A matrix-algebraic algorithm for the Riemannian logarithm on the Stiefel manifold under the canonical metric*, SIAM Journal on Matrix Analysis and Applications **38**:2 (2017), 322–342.
14. R. Zimmermann, A. Vendl, and S. Görtz, *Reduced-order modeling of steady flows subject to aerodynamic constraints*, AIAA Journal **52**:2 (2014), 255–266.